Projets Programmation impérative 1 – langage C

Les projets devront être rendus le 24 janvier 2011, avant 23 heures (heure de Paris).

Ils doivent être réalisés par binômes (C'est-à-dire par groupe de DEUX étudiants) ou à un seul étudiant.

Le code source de votre projet devra être envoyé par email (rchaabane@ai.univ-paris8.fr) avec un rapport (au format PDF) qui devra accompagner les sources dans l'email. Le rapport doit comporter une présentation du travail et des difficultés rencontrées ainsi qu'un listing du programme accompagné d'un mode d'emploi explicite et d'un listing d'exécution.

Vous pouvez poser vos questions ou demander un rendez-vous pour discuter de l'avancement de vos projets par mail à votre enseignant et ceci jusqu'au vendredi 21 janvier inclus.

1. **Lisp**

   Écrire en C un programme capable d'exécuter du Lisp standard. Les programmes Lisp peuvent être :

   - directement écrits en ligne de commande
     Le code est interprété une fois que toutes les parenthèses ouvertes ont été fermées. Ceci permettrait d'écrire des fonctions qui tiennent sur plusieurs lignes sans passer par un éditeur de texte.
   - chargés et exécutés à partir d'un fichier.

   L'interprète doit :

   - avoir une gestion dynamique de la mémoire
   - gérer les fonctions arithmétiques
   - inclure `t`, `nil`, `undefined`, `car`, `cdr`, `cons`, `quote`, `print`, `let`, `if`, `eq`, `atom` et le plus important, **`de`**, ce qui permettra de créer les fonctions manquantes directement en Lisp et les charger au démarrage du programme (ex. : `append`, `length`, `caar`, `cadr`,...)

2. **Labyrinthe 3D : le spéléologue**

   Un labyrinthe peut être vu comme une structure de graphe, c'est-à-dire que, pour chaque position, on connait un pointeur sur les positions disponibles à partir de là. Dans un labyrinthe 3D à salles cubiques, il y a au maximum 6 voisins pour chaque salle. En réalité, il y a une infinité de successeurs possibles. Donc, à partir de chaque salle, il faut disposer d'une liste de salles atteignables directement.

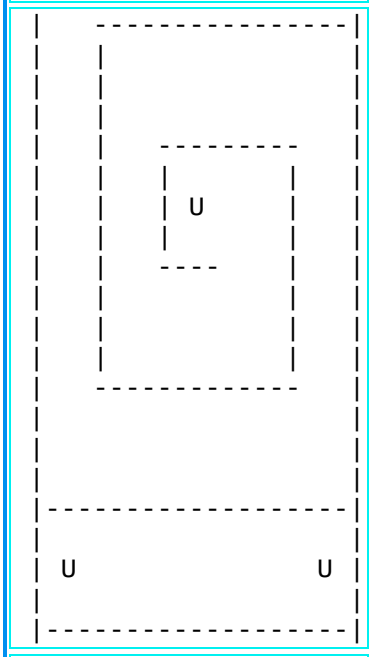
Définir le labyrinthe de la façon la plus simple possible. Programmer un algorithme qui trouve une façon de sortir du labyrinthe quelle que soit la salle dans laquelle on se trouve.

Programmer une méthode permettant de trouver la façon la plus courte de sortir du labyrinthe quelle que soit la salle dans laquelle on se trouve.

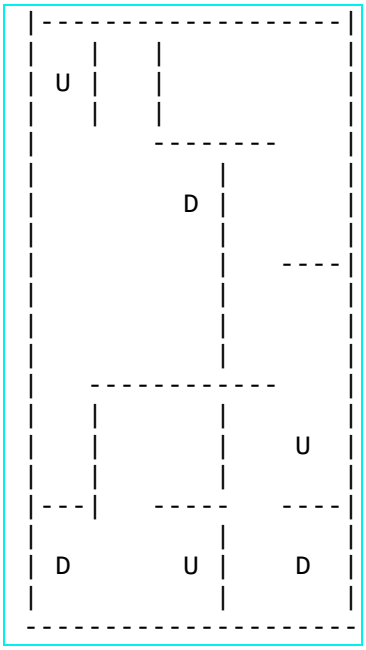
Nous donnons ici un exemple de labyrinthe simple, mais il est possible de faire plus retort.

Exemple Chaque salle est donnée avec U (up) si on peut monter, D (down) si on peut descendre et les quatre (éventuellement) murs.

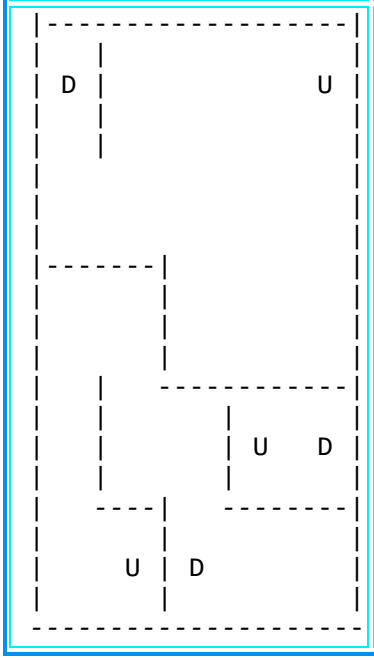
Premier niveau



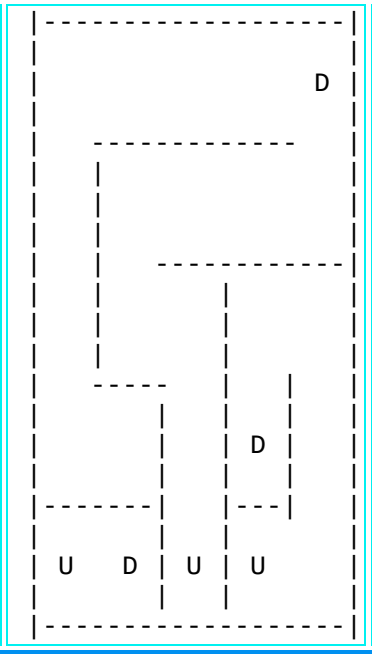
Deuxième niveau



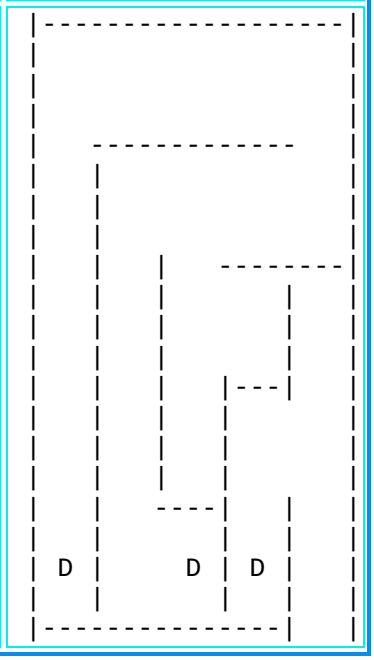
Troisième niveau



Quatrième niveau



Cinquième niveau



3. **Awélé**

   Il s'agit ici de programmer un awélé. Le joueur doit pouvoir jouer contre la machine. On trouvera les règles sur la page suivante.

   http://www.unifr.ch/psycho/pgp/warritournament/reglesFR.pdf

   Il s'agit de pouvoir jouer contre l'ordinateur. Comme, à chaque coup, il y a douze possibilités, il est assez facile de faire un programme qui calcule le meilleur coup. Mais comme au coup suivant, la maigre victoire acquise peut devenir une catastrophe, il vaut mieux calculer les 12 coups qui suivent et même les 144 coups suivants, non ?

4. **Quatro**

   Le quatro est un jeu malgache, qui entre dans la catégorie des awélés, mais en plus riche puisque le plateau contient quatre rangées de huit cases. Chaque joueur tourne sur deux rangées. Chaque fois qu'il "prend", il prend dans la case adjacente de l'adversaire.

   Il faut pouvoir jouer contre la machine.

5. **Morpion solitaire**

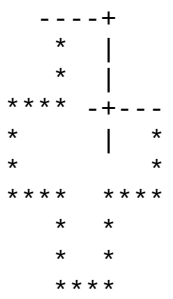
   Jouer au morpion, c'est facile, vous avez tous fait cela. On joue sur une grande grille régulière, par exemple une feuille A4 avec quadrillage 5 mm. Le joueur dessine des croix en cherchant à en aligner des groupes de cinq. http://fr.wikipedia.org/wiki/Morpion_solitaire

   La situation initiale consiste à prendre une page et à dessiner un certain nombre de croix initiales. Puis, à chaque coup, le joueur doit faire un nouvel alignement de cinq croix. Un alignement est donc formé de quatre segments. Un segment ne peut être utilisé qu'une fois. Jusqu'à combien de croix supplémentaires ira votre programme ? 100 ? Position initiale :

Un premier coup pourrait être :

À quoi peut succéder :

puis :



And so on...

6. **Problème du Cheval fou**

Vous connaissez aux échecs le mouvement et le problème du cheval ? Il s'agit de faire parcourir toutes les cases à un cavalier, en suivant les mouvements normaux du cavalier aux échecs, sans jamais repasser par la même case.

Ici, avec un cheval fou, parfois, disons tous les trois coups, il rue et saute, dans un mouvement plus long que d'habitude : au lieu d'un mouvement (2,1) il fait alors un mouvement (3,2). Par exemple, à partir de la case A1, le cheval fou peut passer (premier coup) en B3 (ou en C2) puis (deuxième coup) en C5 (ou C1 ou...) puis, de là, en (troisième coup, le saut) F7 (ou en E2...). And so on. Il s'agit de simplement programmer ces mouvements et faire une fonction qui essaie de faire

bouger le cheval jusqu'à ce que toutes les cases aient été parcourues une et une seule fois chacune.

7. **Quel taquin !**

Le taquin est un jeu dans lequel on déplace des pièces carrées jusqu'à les mettre toutes à leur place réservée. Une adaptation de ce jeu consiste à disposer de pièces de taille variée (par exemple des pièces de taille 1 x 1 et des pièces de taille 1 x 2) et de les amener à leur position.

Exemple :
ici, un numéro signifie une pièce et un numéro répété est celui d'une pièce plus grande. Un vide signifie la case vide.

Situation initiale

```
---------------
| 7  1  3  3  |
| 7  4  4  5  |
| 8    10  9  |
|11  2 12  6  |
---------------
```

Situation à obtenir

```
---------------
| 1  2  3  3  |
| 4  4  5  6  |
| 7  8  9 10  |
| 7 11 12     |
---------------
```

Écrire un programme qui prend une situation initiale, une situation à obtenir et propose les mouvements pour y aboutir.

8. **Nombres Croisés**

Soit une grille de nombres croisés, exemple :

```
    a b c
  ---------
1 |       |
2 |     * |
3 |       |
  ---------
```

1) suite de chiffres consécutifs
2) nombre premier
3) carré
a) carré et palindrome
b) le produit des chiffres est 12
*) case noire

Écrire un programme qui résout ce genre de problèmes. Dans un premier temps on limitera le nombre de définitions (carré, palindrome) et la taille de la grille.

Dans un second temps on ajoutera des styles de définitions ("le carré de la somme des chiffres est un palindrome") et la taille de la grille sera étendue (5x5 puis 6x6).

9. **Anagrammes**

Soit une grille d'anagrammes (c'est-à-dire de mots croisés dont on connaît non pas une définition mais la liste des lettres). On suppose l'existence d'un dictionnaire (qu'il faudra, en partie, construire). Écrire un programme qui trouve la solution de ces anagrammes. Dans un premier temps le dictionnaire sera limité et la taille de la grille aussi. Dans un second temps ces éléments pourront grandir jusqu'à être en mesure de résoudre les anagrammes des journaux spécialisés.

10. **Jeu du 7-0**

Le jeu du 7-0 est assez simple. Chaque joueur (au moins deux joueurs) dispose d'une *cagnotte*, à 0 au départ du jeu. Le but est d'être le joueur dont la cagnotte est la plus grande à la fin d'un tour où une au moins des cagnottes des joueurs a dépassé 200.

Un tour de jeu se passe ainsi, chaque joueur joue successivement. À son tour le joueur lance deux dés à six faces et fait leur somme. Il obtient un total partiel et peut parfois choisir de relancer pour augmenter son total partiel. À un moment où à un autre, il décide de ne pas relancer et de mettre le total partiel dans la cagnotte, c'est alors au joueur suivant. **À chaque lancé de dés, si le total est 7, le total partiel est remis à 0 et le joueur a fini son tour.**

Exemple de début de partie

X joue et fait 4, il rejoue et fait 8 (total partiel 12), il rejoue et fait 11 (total partiel 23), satisfait, il s'arrête et sa cagnotte passe à 23. Y joue et fait 12, il rejoue et fait 11, il rejoue et fait 2, il rejoue et fait 6, il rejoue et fait 10, à ce moment son total partiel est 41, il décide de monter à deux fois le résultat de son adversaire et rejoue. Il fait 7, sa cagnotte reste à 0 et c'est au joueur suivant.

11. Polynômes

Un polynôme est une liste de monômes. On peut le dériver, ou l'intégrer, l'appliquer à une valeur, en additionner, soustraire, multiplier ou diviser deux... Il faut pouvoir enchaîner les calculs, mémoriser les polynômes produits...

12. Matrices

Projet similaire aux polynômes, mais avec des matrices.

13. Calculatrice

Réaliser une calculatrice qui fonctionne en mode texte (non graphique) qui permette d'effectuer une série de calculs (additions, soustractions, multiplications, divisions, log, exponentielle, etc) sur une même ligne. La touche entrée permettra de calculer le résultat.

Vous pourrez programmer des actions liées à des touches claviers tel que les touches de directions (sup, inf) qui permettent d'afficher les 10 dernières opérations de calcul effectuées. Tout comme dans un terminal Shell. Une autre touche (au choix) pour effacer les opérations stockées en mémoire. La touche F1 par exemple pour afficher le mode d'emploi de la calculatrice.

14. Gestion d'une bibliothèque

Ecrire un programme qui permette de gérer les lecteurs et les livres d'une bibliothèque. Ce programme stockera ses données dans des fichiers. Un fichier pour les livres, un fichier pour les lecteurs, et un fichier pour les emprunts.

- Un livre aura des informations qui lui sont liées telles que : titre, auteur(s), année d'édition, et statut (disponible à l'emprunt ou non disponible). Si le statut du livre est non disponible alors on aura sa date d'emprunt, sa date de retour, l'identifiant du lecteur qui l'a emprunté et un statut de retour (en retard, période d'emprunt en cours).
  Si la date de retour est atteinte et que le livre est rendu, le statut du livre passe à « disponible à l'emprunt » et les informations liées à l'emprunt sont supprimées.

- Les lecteurs auront également des informations qui leurs sont liées telles que : identifiant (entier), nom, prénom, âge, adresse, téléphone, liste des livres empruntés, liste des livres en retard.

- Le fichier des emprunts contiendra la liste des livres empruntés, leur statut (en retard ou non en retard) et le numéro du lecteur qui les a empruntés.

Votre programme offrira une interface qui permettra d'afficher :

- La liste des lecteurs
- La liste des livres disponibles
- La liste des livres en retard avec le nom, prénom et téléphone des lecteurs qui les ont empruntés en face de chaque livre. Vous pourrez afficher les livres en retard par lecteur.